Reducing IDS False Positives by Clustering Related Alerts

Mark Heckman

Promia, Inc.

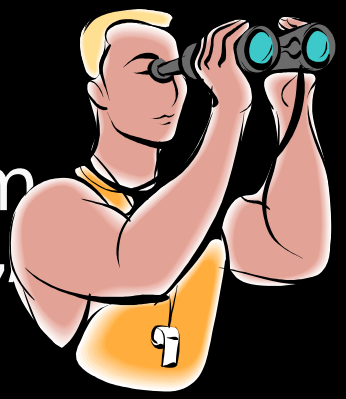
Davis, California

New Research

- Begun in early Spring 2003
- Goal: to detect multistage attacks and reduce false positives
- To be part of the Promia IASM security information management and analysis system
- To be deployed in Fall 2003-Spring 2004

Intrusion Detection Systems

- What an IDS does:
  - monitors a protected system
  - detects “suspicious activity”
  - reports what it detects
- Examples: Cisco IDS (formerly NetRanger), ISS RealSecure, Snort



What is *Suspicious Activity*?

- Activity that has been previously correlated with known attacks (a.k.a. *signatures*)
- Sometimes, simply unusual activity (anomalies)

Problems with IDSs

- Lots of benign activity looks suspicious (a.k.a. *false positives*)
- Some real attacks don’t (a.k.a. *false negatives*)

Why are False Positives Bad?

- Suck up security resources
- Distract/confuse human analysts
- Real attacks are less likely to be detected and stopped
- Ratio of false positives to true positives may be thousands to one

Why not Eliminate False Positives?

- IDSs can only detect suspicious activity
- To eliminate false positives means losing some true positives
- Need human analyst to determine what is a real attack, what is a successful attack, etc.
  - Humans use other tools, domain knowledge, etc.
  - Cannot yet automate what human analysts do

The Security Analyst's Nightmare

- Protecting critical computing resources
- Have Intrusion Detection Systems
- But, a **flood** of IDS alerts
  - Some alerts are true; most are false
- You have limited resources
- The security of the free world depends on you. What do you do?

Today's Topic

- Solutions to the false positive problem
- Focus on *clustering* related alerts
- A simple model for determining *relatedness*
- Implementation using *requires/provides capabilities*

Some Real-world Solutions

- Throw the IDS away
- Put tape over the blinking red light on the console and ignore the alerts
- Massively staff and keep up as well as you can, hoping that, as traffic grows lighter overnight, you can catch up before the next morning

A Better Solution

- Identify most common false alerts
- Turn them off (configure IDS)
- Reduce analyst’s workload

- But, may be blind to real attack!

The “Best” Practical Solution

- Prioritize alerts
- Focus on the “most interesting”
  - Dangerous (vulnerable or critical resource)
  - Unusual (anomalous)
  - Organized (related in a pattern of directed, intelligent activity)

Prioritizing Based on Vulnerability

- E.g., if you have Apache web server, ignore IIS attacks
  - Or ignore if old attack and you are patched
- But, need database of versions, vulnerabilities
- May miss warnings and precursors

Prioritizing Based on Anomalousness

- Real attacks are likely to be unusual when compared to normal activity
- But, unusual not necessarily bad
  - E.g., host-based user monitors, Bro project
- Normal not necessarily good
  - E.g., Nimda
- Usually, no semantics
  - E.g., Therminator project (?)

Prioritizing Based on Organization

- *Organized* alerts more likely to be real
  - *Organized* means related in a pattern of directed, intelligent activity – a model
- E.g., Attacks usually occur in steps
  - Ping sweep, port scan, exploit
- But, will miss “magic bullet” attacks
- Will flag benign, directed activity

Ideally, Combine all Three Methods

- Each compensates for weaknesses of others
- E.g., Vulnerabilities and Organization give semantics that Anomalousness lacks
- Next stage in IDS development?

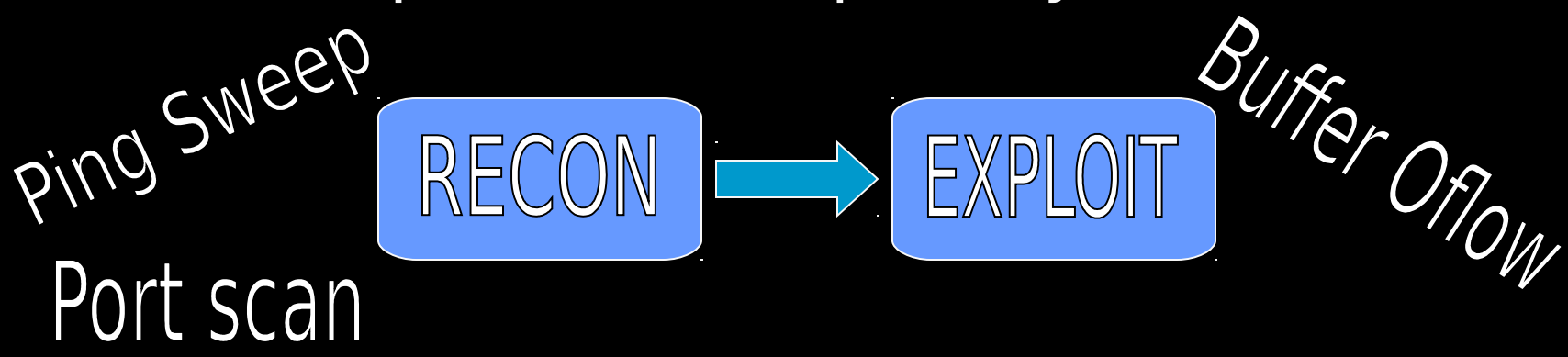
How to Identify Organized Alerts (1)

- Simple pattern: Match alerts by source and destination
  - Almost no semantics

- Intermediate pattern: Group alerts that are known to represent specific attacks (*meta-signatures*)
  - Will miss complex or new attacks

How to Identify Organized Alerts (2)

- Advanced Intermediate pattern: Use a *model* of attack stages, assign alerts to stages
  - Attack semantics
  - Basic predictive capability



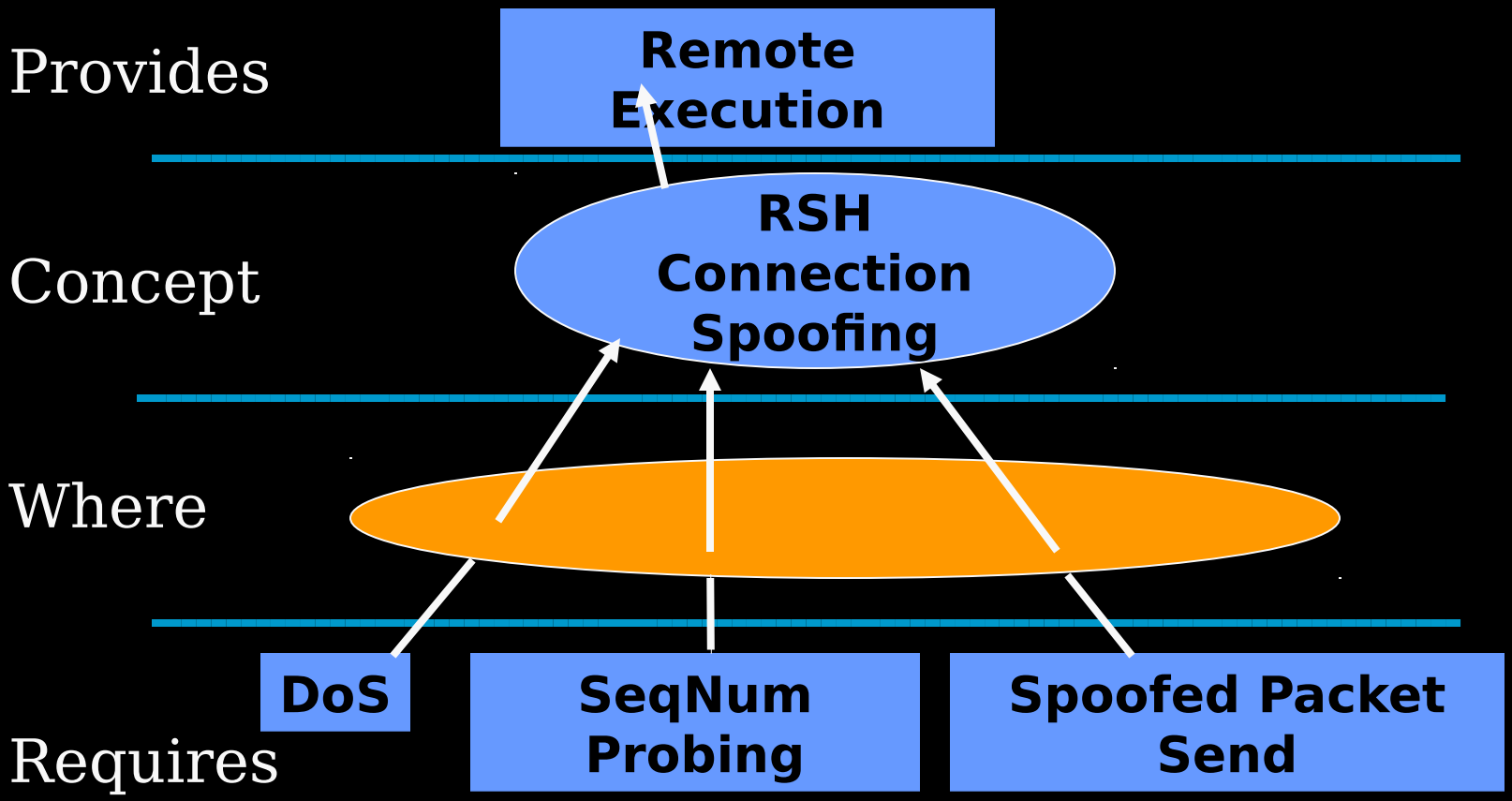
How to Identify Organized Alerts (3)

- Advanced pattern: Use a model of attacker *capabilities*
- Each alert indicates new capabilities gained by attacker
- One attack step provides new capabilities that are required by another attack step

Capability Model of Attacks

- Per Templeton and Levitt, 2000
- Model of attacks based on abstract *concepts*
- A concept requires certain *capabilities*,
  - specific information or a specific situation that must hold for the concept to hold
- A concept may, in turn, provide capabilities to another concept

Concept Structure



Model Pros and Cons

- It is hierarchical and abstract
- Does not require a priori knowledge of particular scenario so can describe many, previously unseen, scenarios
- But, must invent concepts – is the model ever complete?
- May not have sensors for some concepts

Applying Capability Model to IDSs

- Want to use IDSs as sensors to capability model
- IDS alerts indicate capabilities
- Concepts whose capabilities can’t be determined through IDS alerts can’t be used (or can be used only partly)
- Some alerts might not correspond to any concepts if model is incomplete

A Brilliant Insight

- Signature-based IDSs constitute an implicit attack model
- Each alert description contains some idea of the requires and provides
- Each alert is a concept
- Complete set of alerts (concepts) is a “complete” model of attacks

Example Alert Description

- Alert: IP options-Strict Source Route
- Description: “Setting this option can allow a remote host to pose as a local host on your network. Services that rely on IP addresses as authentication may be compromised as a result.”

Possible Concept for this Alert

- Concept: IP options-Strict Source Route
- Requires:
  - Attacker knows target exists
  - Network path between attacker and target
  - Attacker knows route to target
- Provides:
  - Identity Masquerade

OK, Maybe not so Brilliant an Insight

- An alert usually only a partial concept
  - E.g., example doesn't include other capabilities necessary for spoof to work
- Describes suspicious activity, but not certain if provides are ever true
  - E.g., "can allow a remote host", or "may be compromised"
- Hard to infer the capabilities

So, It’s not Perfect; Deal with it

- Even a partial concept is acceptable – use whatever capabilities are suggested
  - Other alerts may fill in missing capabilities
- Assume the worst case, that all the “cans” and “maybes” happen
  - If don’t happen, then won’t be any follow-on alerts
  - May have other sensors that can tell

But, what about the Capabilities?

- Recipe:
  - Look at the hundreds of alerts over and over again
  - Induce a model
- Signature writers didn’t have abstract model in mind, but it is there

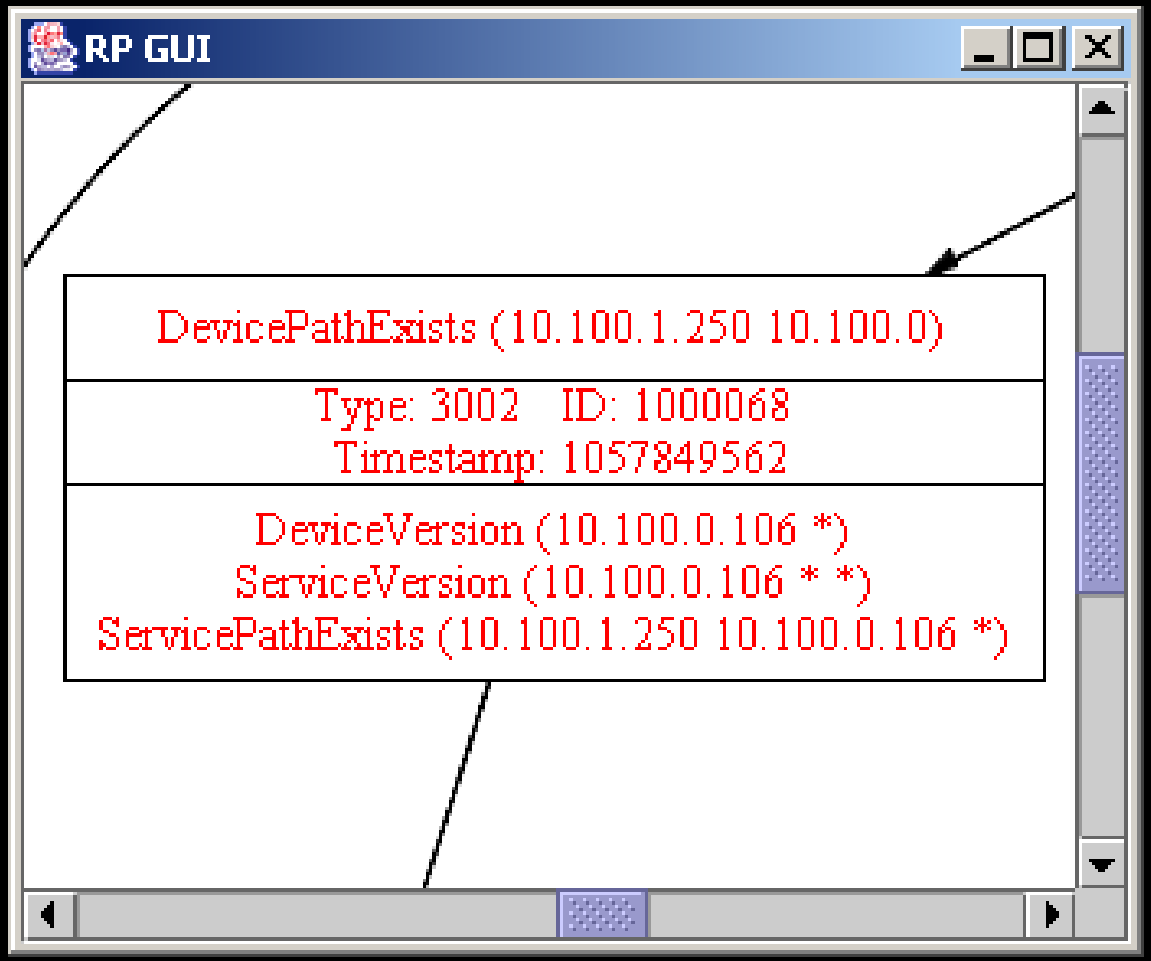
Basis for an IDS Capability Model

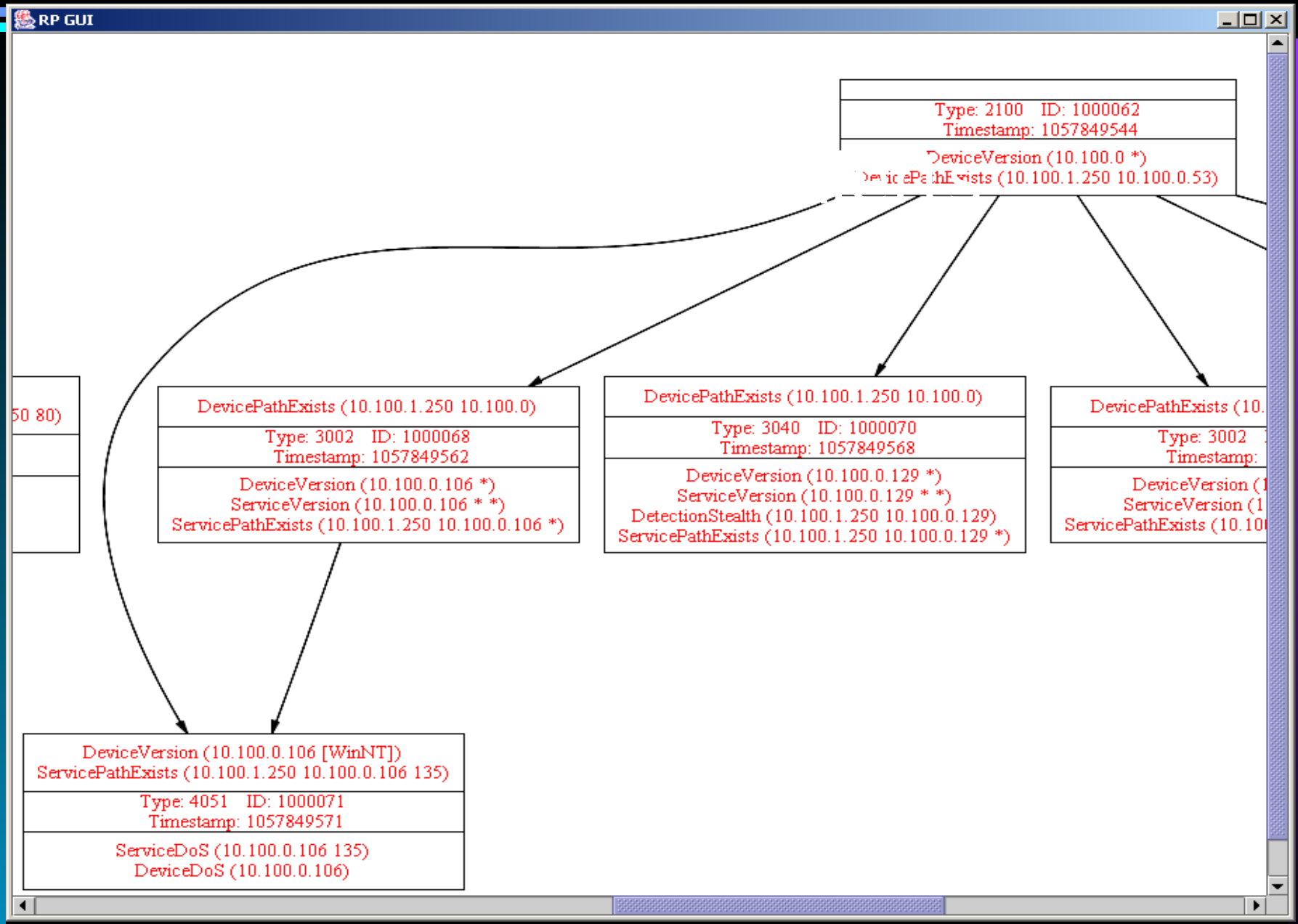
- Objects
  - Devices, services, etc.
- Attributes (and Attack Types)
  - Existence (recon), Version (recon), Function (DoS), etc.
- (Object,Attribute) -> Capability
  - E.g., DeviceExists, DeviceVersion, DeviceDoS, ServiceDoS, etc.

Promia IDS Capability Model

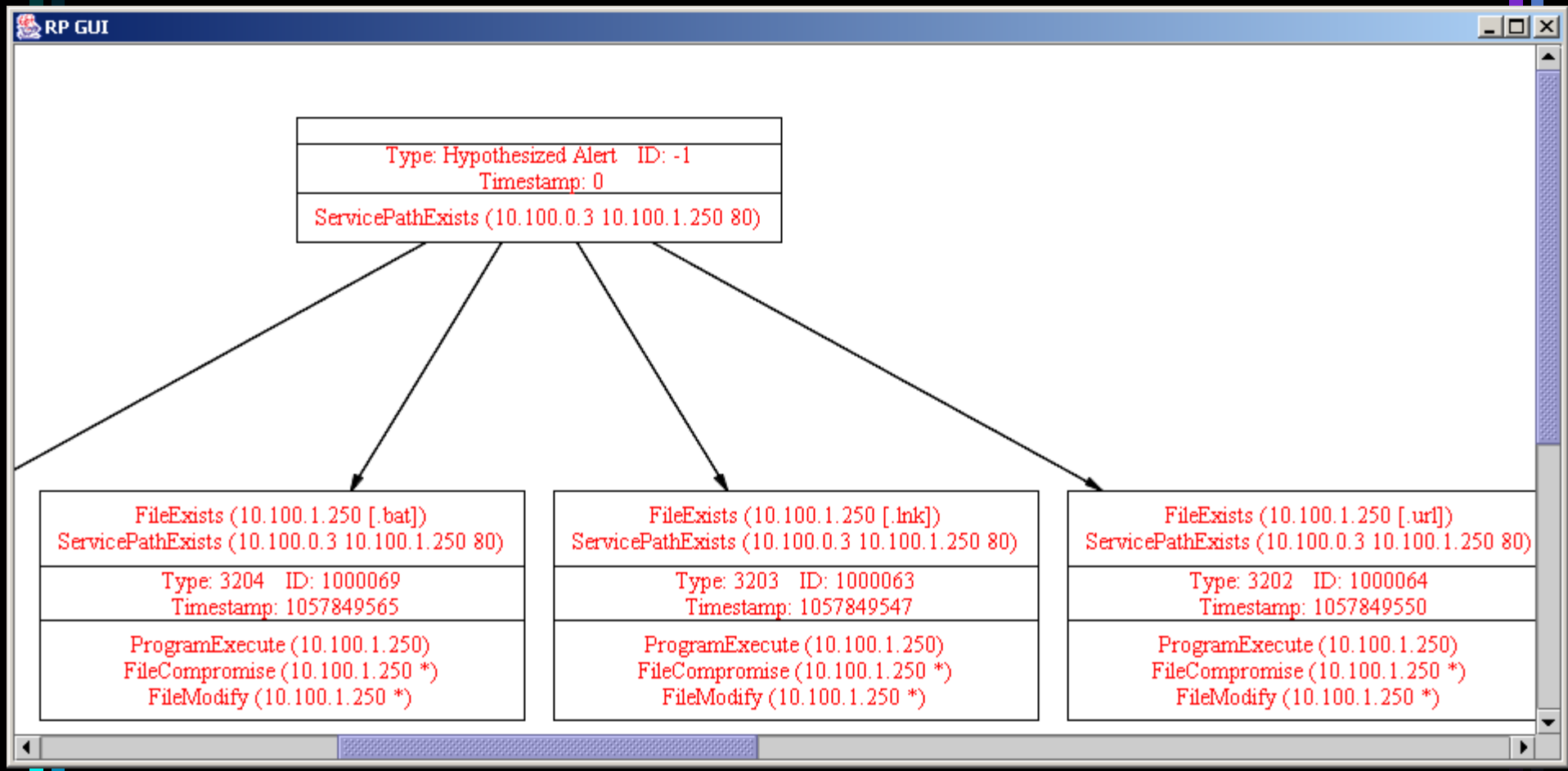
- First done for Cisco IDS (network-based) - Currently, 20-25 capabilities
- Coming soon: RealSecure and Snort
  - RealSecure has some host-based alerts and so will require new capabilities
- Will add OS logs, firewalls, and other sensors

An Early Prototype




RP GUI
Type: 2100 ID: 1000062
Timestamp: 1057849544
DeviceVersion (10.100.0 *)
DevicePathExists (10.100.1.250 10.100.0.53)
50 80)
DevicePathExists (10.100.1.250 10.100.0)
Type: 3002 ID: 1000068
Timestamp: 1057849562
DeviceVersion (10.100.0.106 *)
ServiceVersion (10.100.0.106 * *)
ServicePathExists (10.100.1.250 10.100.0.106 *)
DevicePathExists (10.100.1.250 10.100.0)
Type: 3040 ID: 1000070
Timestamp: 1057849568
DeviceVersion (10.100.0.129 *)
ServiceVersion (10.100.0.129 * *)
DetectionStealth (10.100.1.250 10.100.0.129)
ServicePathExists (10.100.1.250 10.100.0.129 *)
DevicePathExists (10.
Type: 3002
Timestamp:
DeviceVersion (
ServiceVersion (1
ServicePathExists (10.10
DeviceVersion (10.100.0.106 [WinNT])
ServicePathExists (10.100.1.250 10.100.0.106 135)
Type: 4051 ID: 1000071
Timestamp: 1057849571
ServiceDoS (10.100.0.106 135)
DeviceDoS (10.100.0.106)

Hypothesized Concepts



Problem with Managing Alerts

- In a busy installation, may have thousands of alerts every hour
- A real attack, however, may manifest itself over many hours or days
- But, r/p system has limited memory and other resources, so can’t keep all the alerts around for potential clustering

Solutions to Managing Alerts Problem

- Aggregate identical or similar alerts
  - Keep counts; throw away the duplicates
- Use anomaly values to determine how long to keep an alert around
  - Real attacks more likely to be unusual
- Run r/p clustering batch on archived data
  - E.g., slice based on target IP

Lessons Learned So Far

- Using IDS alerts as concepts, it is hard to come up with good, complete set of capabilities
- Hard to consistently apply capabilities to alerts/concepts
  - Need standard key/guide
- But, better than starting from scratch

Things I Didn't Talk About, but Someone Else is Working on Them

- A formal model of how to link alerts using requires/provides capabilities
  - Cuppens and Miege, 2002
- How to make analytical use of the alert clusters
  - Peng Ning, et. al., 2001

Contact Info

Mark Heckman
Promia, Inc.
1490 Drew Ave., Ste. 180
Davis, CA 95616

530-756-2598 (temporary)
mheckman@promia.com